Remarks

The above amendments and these remarks are in reply to the Office Action mailed October 31, 2008 and the Advisory Action mailed February 2, 2009.

I. Summary of Examiner's Rejections

In the Advisory Action mailed February 2, 2009, Claims 1, 2, 5-9, 12, 13, 16-20, 29, and 30 stood rejected to under 35 U.S.C. 103(a) as allegedly unpatentable over Kampe et al. (U.S. Pat. No. 6,854,069, hereinafter Kampe).

II. Summary of Applicants' Response

The present Reply amends claims 1, 12, and 29; leaving for the Examiner's present consideration claims 1, 2, 2, 5-9, 12, 13, 16-20, 29, and 30. Reconsideration of the claims is requested.

III. Rejections under 35 U.S.C. 103(a)

In the Advisory Action mailed February 2, 2009, Claims 1, 2, 5-9, 12, 13, 16-20, 29, and 30 stood rejected to under 35 U.S.C. 103(a) as allegedly unpatentable over Kampe et al. (U.S. Pat. No. 6,854,069, hereinafter Kampe).

Claim 1

Claim 1 has been amended to more clearly amend the embodiment therein. As amended, Claim 1 currently defines:

> *1. A system for high availability clustering of a group of computer nodes, comprising:*
>
> *a Java-based cluster server executing on a Java virtual machine on a computer, wherein said Java-based cluster server provides an application access to a set of resources of multiple resource types, wherein two or more resource types correspond to two or more different application servers within a cluster, wherein said resources, and application servers are available at one or more computers in the cluster, and wherein the resources, and application servers are grouped by resource type within the set of resources;*

a resource interface provided by said Java-based cluster server that provides an abstraction layer and allows the Java-based cluster server to receive uniform requests from the application and communicate the requests to said set of resources;

a plurality of plugins that are plugged into the resource interface to provide a set of application-specific callbacks from the Java-based cluster server to the set of resources, wherein the system includes a plugin for each resource type corresponding to the different application server, and wherein each plugin implements a resource API to encapsulate the plugin's particular resource type-specific behavior and to isolate the Java-based cluster server from said behavior while providing access to its pool of resources; and

a JNDI interface provided by said Java-based cluster server, wherein the JNDI interface provides an interface between the Java-based cluster server and a JNDI-compliant database;

wherein the resource interface accepts additional plugins that are plugged into the resource interface for other resource types; and

a GLobal Update Protocol (GLUP) mechanism that employs a distributed global lock with sequence numbers to serialize propagation of global events across active members of the cluster, wherein each global update is associated with a unique sequence number such that each said active member of the cluster has an identical view of an ordering of the global events.

As amended, Claim 1 now specifically defines that the JVM-based cluster server includes a plurality of plugins where the plugins correspond to multiple resource types. There are two or more resource types and they correspond to two or more different application servers. Therefore, for each different application server within the cluster, there exists a different plugin associated with it.

In addition, Claim 1 has been amended to include a GLobal Update Protocol (GLUP) mechanism that employs a distributed lock with sequence numbers to serialize propagation of global events across active members of the cluster. Specifically, each global update is associated with its own unique sequence number such that each active member in the cluster is provided with an identical view of the ordering of the global events (Specification, par. [0060]).

The cited reference Kampe teaches a method and system for achieving high availability in a networked computer system. More particularly, Kampe appears to describe using high-availability-aware components to represent hardware and software in the system. These

components are managed to achieve a level or redundancy. Furthermore, the health of the computer system (the nodes) is monitored (Kampe, Abstract).

However, Applicant respectfully submits that Kampe fails to disclose or render obvious the features of Claim 1, as amended.

Specifically, Kampe fails to disclose that a JVM-based cluster server includes plugins that are associated with different application server types, as defined in amended Claim 1. As pointed out in the Office Action, Kampe does mention some "plug-in components (e.g. device drivers, protocols, applications, etc.) to facilitate the addition of new components to a system." (Kampe, col. 5, lines 35-45). However, there is no mention of multiple different application servers that are associated with the plugins. Furthermore, Kampe does not appear to disclose a plurality of plugins that are plugged into a resource interface to provide a set of application-specific callbacks from a Java-based cluster server to a set of resources, as defined by claim 1.

Furthermore, Kampe fails to disclose *a GLobal Update Protocol (GLUP) mechanism that employs a distributed global lock with sequence numbers to serialize propagation of global events across active members of the cluster, wherein each global update is associated with a unique sequence number such that each said active member of the cluster has an identical view of an ordering of the global events*, as defined in amended Claim 1. This feature of Claim 1 synchronizes and coordinates the global events across the members of the cluster. Kampe does not assign any sequence numbers to any global events. Moreover, Kampe is silent regarding each active member of the cluster having an identical view of the ordering of the global events.

In view of the comments provided above, Applicants respectfully submit that the embodiment defined by claim 1, as amended, is neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is respectfully requested.

Claims 12 and 29

The comments provided above with respect to claim 1 are hereby incorporated by reference. Claims 12 and 29 have been similarly amended by the current Reply to more clearly define the embodiments therein. For similar reasons as provided above with respect to Claim 1, Applicants respectfully submit that Claims 12 and 29, as amended, are likewise neither anticipated by, nor obvious in view of the cited references, and reconsideration thereof is

respectfully requested.

Claims 2, 5-9, 13, 16-20, and 30

Claims 2, 5-9, 13, 16-20, and 30 depend from and include all of the features of Claims 1, 12, or 29. Claims 2, 5-9, 13, 16-20, and 30 are not addressed separately, but it is respectfully submitted that these claims are allowable as depending from an allowable independent claim, and further in view of the amendments to the independent claims, and the comments provided above. Reconsideration thereof is respectfully requested.

IV. Conclusion

In view of the above amendments and remarks, it is respectfully submitted that all of the claims now pending in the subject patent application should be allowable, and reconsideration thereof is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if he can assist in any way in expediting issuance of a patent.

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: March 31, 2009

By: /Justas Geringson/
Justas Geringson
Reg. No. 57,033

Customer No.: 80548
FLIESLER MEYER LLP
650 California Street, 14th Floor
San Francisco, California 94108
Telephone: (415) 362-3800